





Updates to update-all-versions p2z branch

G. Cerati (FNAL) p2z/p2r Meeting June 13, 2023

Fit failures

- Current smearing value is 10⁻⁵ (where units depend on parameter).
- No physical meaning in smearing values, goal is just to avoid having identical results
- Reducing smearing by factor 10-100x eliminates nans and fit failures
 - with 10⁻⁶ we only have one residual failure when using icc, which goes away when using "-fp-model strict" in compilation
- I'd be in favor of reducing smearing and getting rid of failures so that we do not need to discuss them in the paper



Fix to kokkos v3

bug affecting output - may be there also for other GPU versions

#pragma omp simd

Kokkos v3

```
Kokkos::parallel_for( Kokkos::TeamVectorRange(teamMember, bsize),[&](const size_t it)
   newErr->data[ 0*bsize+it] = kGain->data[ 0*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[ 1*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[ 1*bsize+it] = kGain->data[ 2*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[ 3*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[ 2*bsize+it] = kGain->data[ 2*bsize+it]*trkErr->data[ 1*bsize+it] + kGain->data[ 3*bsize+it]*trkErr->data[ 2*bsize+it];
   newErr->data[ 3*bsize+it] = kGain->data[ 4*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[ 5*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[ 4*bsize+it] = kGain->data[ 4*bsize+it]*trkErr->data[ 1*bsize+it] + kGain->data[ 5*bsize+it]*trkErr->data[ 2*bsize+it];
   newErr->data[ 5*bsize+it] = kGain->data[ 4*bsize+it]*trkErr->data[ 3*bsize+it] + kGain->data[ 5*bsize+it]*trkErr->data[ 4*bsize+it];
   newErr->data[ 6*bsize+it] = kGain->data[ 6*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[ 7*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[ 7*bsize+it] = kGain->data[ 6*bsize+it]*trkErr->data[ 1*bsize+it] + kGain->data[ 7*bsize+it]*trkErr->data[ 2*bsize+it];
   newErr->data[ 8*bsize+it] = kGain->data[ 6*bsize+it]*trkErr->data[ 3*bsize+it] + kGain->data[ 7*bsize+it]*trkErr->data[ 4*bsize+it];
   newErr->data[ 9*bsize+it] = kGain->data[ 6*bsize+it]*trkErr->data[ 6*bsize+it] + kGain->data[ 7*bsize+it]*trkErr->data[ 7*bsize+it];
   newErr->data[10*bsize+it] = kGain->data[ 8*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[ 9*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[11*bsize+it] = kGain->data[ 8*bsize+it]*trkErr->data[ 1*bsize+it] + kGain->data[ 9*bsize+it]*trkErr->data[ 2*bsize+it];
   newErr->data[12*bsize+it] = kGain->data[ 8*bsize+it]*trkErr->data[ 3*bsize+it] + kGain->data[ 9*bsize+it]*trkErr->data[ 4*bsize+it];
   newErr->data[13*bsize+it] = kGain->data[ 8*bsize+it]*trkErr->data[ 6*bsize+it] + kGain->data[ 9*bsize+it]*trkErr->data[ 7*bsize+it];
   newErr->data[14*bsize+it] = kGain->data[ 8*bsize+it]*trkErr->data[10*bsize+it] + kGain->data[ 9*bsize+it]*trkErr->data[11*bsize+it];
   newErr->data[15*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[ 0*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[ 1*bsize+it];
   newErr->data[16*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[ 1*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[ 2*bsize+it];
   newErr->data[17*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[ 3*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[ 4*bsize+it];
   newErr->data[18*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[6*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[7*bsize+it];
   newErr->data[19*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[10*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[11*bsize+it];
   newErr->data[20*bsize+it] = kGain->data[10*bsize+it]*trkErr->data[15*bsize+it] + kGain->data[11*bsize+it]*trkErr->data[16*bsize+it];
```

```
newErr->data[ 0*bsize+it] = trkErr->data[ 0*bsize+it] - newErr->data[ 0*bsize+it];
newErr->data[ 1*bsize+it] = trkErr->data[ 1*bsize+it] - newErr->data[ 1*bsize+it];
newErr->data[ 2*bsize+it] = trkErr->data[ 2*bsize+it] - newErr->data[ 2*bsize+it];
newErr->data[ 3*bsize+it] = trkErr->data[ 3*bsize+it] - newErr->data[ 3*bsize+it];
newErr->data[ 4*bsize+it] = trkErr->data[ 4*bsize+it] - newErr->data[ 4*bsize+it];
newErr->data[ 5*bsize+it] = trkErr->data[ 5*bsize+it] - newErr->data[ 5*bsize+it];
newErr->data[ 6*bsize+it] = trkErr->data[ 6*bsize+it] - newErr->data[ 6*bsize+it];
newErr->data[ 7*bsize+it] = trkErr->data[ 7*bsize+it] - newErr->data[ 7*bsize+it];
newErr->data[ 8*bsize+it] = trkErr->data[ 8*bsize+it] - newErr->data[ 8*bsize+it];
newErr->data[ 9*bsize+it] = trkErr->data[ 9*bsize+it] - newErr->data[ 9*bsize+it];
newErr->data[10*bsize+it] = trkErr->data[10*bsize+it] - newErr->data[10*bsize+it];
newErr->data[11*bsize+it] = trkErr->data[11*bsize+it] - newErr->data[11*bsize+it];
newErr->data[12*bsize+it] = trkErr->data[12*bsize+it] - newErr->data[12*bsize+it];
newErr->data[13*bsize+it] = trkErr->data[13*bsize+it] - newErr->data[13*bsize+it];
newErr->data[14*bsize+it] = trkErr->data[14*bsize+it] - newErr->data[14*bsize+it];
newErr->data[15*bsize+it] = trkErr->data[15*bsize+it] - newErr->data[15*bsize+it];
newErr->data[16*bsize+it] = trkErr->data[16*bsize+it] - newErr->data[16*bsize+it];
newErr->data[17*bsize+it] = trkErr->data[17*bsize+it] - newErr->data[17*bsize+it];
newErr->data[18*bsize+it] = trkErr->data[18*bsize+it] - newErr->data[18*bsize+it];
newErr->data[19*bsize+it] = trkErr->data[19*bsize+it] - newErr->data[19*bsize+it];
newErr->data[20*bsize+it] = trkErr->data[20*bsize+it] - newErr->data[20*bsize+it];
```

removed!
this is already done
in the next loop

```
Kokkos::parallel_for( Kokkos::TeamVectorRange(teamMember, bsize),[&](const size_t it)
{
    #pragma unroll
    for (int i = 0; i < 21; i++){
        trkErr->data[ i*bsize+it] = trkErr->data[ i*bsize+it] - newErr->data[ i*bsize+it];
}
```

TBB

```
for (size_t it=0;it<bsize;++it)</pre>
  newErr.data[ 0*bsize+it] = kGain.data[ 0*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[ 1*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[ 1*bsize+it] = kGain.data[ 2*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[ 3*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[ 2*bsize+it] = kGain.data[ 2*bsize+it]*trkErr->data[ 1*bsize+it] + kGain.data[ 3*bsize+it]*trkErr->data[ 2*bsize+it];
  newErr.data[ 3*bsize+it] = kGain.data[ 4*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[ 5*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[ 4*bsize+it] = kGain.data[ 4*bsize+it]*trkErr->data[ 1*bsize+it] + kGain.data[ 5*bsize+it]*trkErr->data[ 2*bsize+it];
  newErr.data[ 5*bsize+it] = kGain.data[ 4*bsize+it]*trkErr->data[ 3*bsize+it] + kGain.data[ 5*bsize+it]*trkErr->data[ 4*bsize+it];
  newErr.data[ 6*bsize+it] = kGain.data[ 6*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[ 7*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[ 7*bsize+it] = kGain.data[ 6*bsize+it]*trkErr->data[ 1*bsize+it] + kGain.data[ 7*bsize+it]*trkErr->data[ 2*bsize+it];
  newErr.data[ 8*bsize+it] = kGain.data[ 6*bsize+it]*trkErr->data[ 3*bsize+it] + kGain.data[ 7*bsize+it]*trkErr->data[ 4*bsize+it];
  newErr.data[ 9*bsize+it] = kGain.data[ 6*bsize+it]*trkErr->data[ 6*bsize+it] + kGain.data[ 7*bsize+it]*trkErr->data[ 7*bsize+it];
  newErr.data[10*bsize+it] = kGain.data[ 8*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[ 9*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[11*bsize+it] = kGain.data[ 8*bsize+it]*trkErr->data[ 1*bsize+it] + kGain.data[ 9*bsize+it]*trkErr->data[ 2*bsize+it];
  newErr.data[12*bsize+it] = kGain.data[ 8*bsize+it]*trkErr->data[ 3*bsize+it] + kGain.data[ 9*bsize+it]*trkErr->data[ 4*bsize+it];
  newErr.data[13*bsize+it] = kGain.data[ 8*bsize+it]*trkErr->data[ 6*bsize+it] + kGain.data[ 9*bsize+it]*trkErr->data[ 7*bsize+it];
  newErr.data[14*bsize+it] = kGain.data[ 8*bsize+it]*trkErr->data[10*bsize+it] + kGain.data[ 9*bsize+it]*trkErr->data[11*bsize+it];
  newErr.data[15*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[ 0*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[ 1*bsize+it];
  newErr.data[16*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[ 1*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[ 2*bsize+it];
  newErr.data[17*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[ 3*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[ 4*bsize+it];
  newErr.data[18*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[6*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[7*bsize+it];
  newErr.data[19*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[10*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[11*bsize+it];
  newErr.data[20*bsize+it] = kGain.data[10*bsize+it]*trkErr->data[15*bsize+it] + kGain.data[11*bsize+it]*trkErr->data[16*bsize+it];
  newErr.data[ 0*bsize+it] = trkErr->data[ 0*bsize+it] - newErr.data[ 0*bsize+it];
  newErr.data[ 1*bsize+it] = trkErr->data[ 1*bsize+it] - newErr.data[ 1*bsize+it];
  newErr.data[ 2*bsize+it] = trkErr->data[ 2*bsize+it] - newErr.data[ 2*bsize+it];
  newErr.data[ 3*bsize+it] = trkErr->data[ 3*bsize+it] - newErr.data[ 3*bsize+it];
  newErr.data[ 4*bsize+it] = trkErr->data[ 4*bsize+it] - newErr.data[ 4*bsize+it];
  newErr.data[ 5*bsize+it] = trkErr->data[ 5*bsize+it] - newErr.data[ 5*bsize+it];
  newErr.data[ 6*bsize+it] = trkErr->data[ 6*bsize+it] - newErr.data[ 6*bsize+it];
  newErr.data[ 7*bsize+it] = trkErr->data[ 7*bsize+it] - newErr.data[ 7*bsize+it];
  newErr.data[ 8*bsize+it] = trkErr->data[ 8*bsize+it] - newErr.data[ 8*bsize+it];
  newErr.data[ 9*bsize+it] = trkErr->data[ 9*bsize+it] - newErr.data[ 9*bsize+it];
  newErr.data[10*bsize+it] = trkErr->data[10*bsize+it] - newErr.data[10*bsize+it];
  newErr.data[11*bsize+it] = trkErr->data[11*bsize+it] - newErr.data[11*bsize+it];
  newErr.data[12*bsize+it] = trkErr->data[12*bsize+it] - newErr.data[12*bsize+it];
  newErr.data[13*bsize+it] = trkErr->data[13*bsize+it] - newErr.data[13*bsize+it];
  newErr.data[14*bsize+it] = trkErr->data[14*bsize+it] - newErr.data[14*bsize+it];
  newErr.data[15*bsize+it] = trkErr->data[15*bsize+it] - newErr.data[15*bsize+it];
  newErr.data[16*bsize+it] = trkErr->data[16*bsize+it] - newErr.data[16*bsize+it];
  newErr.data[17*bsize+it] = trkErr->data[17*bsize+it] - newErr.data[17*bsize+it];
  newErr.data[18*bsize+it] = trkErr->data[18*bsize+it] - newErr.data[18*bsize+it];
   newErr.data[19*bsize+it] = trkErr->data[19*bsize+it] - newErr.data[19*bsize+it];
  newErr.data[20*bsize+it] = trkErr->data[20*bsize+it] - newErr.data[20*bsize+it];
(*trkErr) = newErr;
```



More uniform output reporting

 Using same conditions and same floating point precision helps getting same output reports

alpaka

```
if (std::isfinite(x_)==false ||
              if (isnan(x_) ||
1114 -
                                                                                          1118 +
1115 -
                                                                                          1119 +
                   isnan(y_) ||
                                                                                                             std::isfinite(y_)==false ||
                                                                                                            std::isfinite(z_)==false ||
1116 -
                   isnan(z_) ||
                                                                                          1120 +
1117 -
                   isnan(pt_) ||
                                                                                          1121 +
                                                                                                             std::isfinite(pt_)==false ||
1118 -
                                                                                                             std::isfinite(phi_)==false ||
                   isnan(phi_) ||
                                                                                          1122 +
1119 -
                   isnan(theta_)
                                                                                          1123 +
                                                                                                             std::isfinite(theta_)==false
                  ) {
1120
                                                                                          1124
1121
                nnans++;
                                                                                          1125
                                                                                                           nnans++;
1122
                                                                                          1126
                 continue;
                                                                                                           continue;
1123
                                                                                          1127
1124
              if (fabs( (x_-hx_)/hx_ )>1. ||
                                                                                          1128
                                                                                                         if (fabs( (x_-hx_)/hx_ )>1. ||
1125
                   fabs( (y_-hy_)/hy_ )>1. ||
                                                                                          1129
                                                                                                             fabs( (y_-hy_)/hy_ )>1. ||
                                                                                                            fabs( (z_-hz_)/hz_ )>1. ||
                  fabs( (z_-hz_)/hz_ )>1. ||
1126
                                                                                          1130
1127 -
                  fabs( (pt_-12.)/12.)>1.
                                                                                          1131 +
                                                                                                             fabs( (pt_-12.)/12.)>1. ||
1128 -
                  ) {
                                                                                          1132 +
                                                                                                             fabs( (phi_-1.3)/1.3)>1. ||
1129 -
                nfail++;
                                                                                          1133 +
                                                                                                             fabs( (theta_-2.8)/2.8)>1.
1130 -
                 continue;
                                                                                          1134 +
                                                                                                           ) {
                                                                                                           nfail++;
                                                                                          1135 +
                                                                                          1136 +
                                                                                                           continue;
```

TBB

```
100
745
                                                                                         784
                                                                                                    int nnans = 0, nfail = 0, ngood = 0;
          int nnans = 0, nfail = 0, ngood = 0;
746 -
          float avgx = 0, avgy = 0, avgz = 0;
                                                                                         785 +
                                                                                                    double avgx = 0, avgy = 0, avgz = 0;
          float avgpt = 0, avgphi = 0, avgtheta = 0;
747 -
                                                                                         786 +
                                                                                                    double avgpt = 0, avgphi = 0, avgtheta = 0;
          float avgdx = 0, avgdy = 0, avgdz = 0;
                                                                                                    double avgdx = 0, avgdy = 0, avgdz = 0;
748 -
                                                                                         787 +
749
          for (size_t ie=0;ie<nevts;++ie) {</pre>
                                                                                                    for (size_t ie=0;ie<nevts;++ie) {</pre>
                                                                                         788
           for (size_t it=0;it<ntrks;++it) {</pre>
750
                                                                                         789
                                                                                                      for (size_t it=0;it<ntrks;++it) {</pre>
751 -
                                                                                         790 +
             float x_ = x(outtrk,ie,it);
                                                                                                       double x_ = x(outtrk,ie,it);
752 -
                                                                                         791 +
             float y_ = y(outtrk,ie,it);
                                                                                                       double y_ = y(outtrk,ie,it);
753 -
                                                                                         792 +
                                                                                                       double z_ = z(outtrk,ie,it);
             float z_ = z(outtrk,ie,it);
754 -
             float pt_ = 1./ipt(outtrk,ie,it);
                                                                                         793 +
                                                                                                       //printf("e=%li t=%li z=%10f \n", ie, it, z_);
755 -
                                                                                         794 +
                                                                                                       double pt_ = 1./ipt(outtrk,ie,it);
              float phi_ = phi(outtrk,ie,it);
756 -
             float theta_ = theta(outtrk,ie,it);
                                                                                         795 +
                                                                                                       double phi_ = phi(outtrk,ie,it);
              float hx_ = inputhits[nlayer-1].pos[0];
                                                                                                       double theta_ = theta(outtrk,ie,it);
                                                                                         796 +
              float hy_ = inputhits[nlayer-1].pos[1];
                                                                                         797 +
                                                                                                       double hx_ = inputhits[nlayer-1].pos[0];
             float hz_ = inputhits[nlayer-1].pos[2];
                                                                                                       double hy_ = inputhits[nlayer-1].pos[1];
                                                                                         799 🛨+
              float hr_ = sqrtf(hx_*hx_ + hy_*hy_);
                                                                                                       double hz_ = inputhits[nlayer-1].pos[2];
760 -
                                                                                         800 +
                                                                                                       double hr_ = sqrtf(hx_*hx_ + hy_*hy_);
                                                                                                       if (std::isfinite(x_)==false ||
761
             if (std::isfinite(x_)==false ||
                                                                                         801
                                                                                                           Fermilab
762
                  std::isfinite(v )==false || |
                                                                                         802
```

Output results - GCC

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_gcc_tbb.log
produce nevts=100 ntrks=9600 smearing by=1.0e-07
NITER=10
done preparing!
setup time time=4.409000 (s)
done ntracks=9600000 tot time=0.554024 (s) time/trk=5.771083e-08 (s)
formatted 10 100 9600 32 300 0.554024 0 4.409000 64
track x avg=6.443559 std/avg=0.000039
track y avg=24.394711 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328098
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
number of tracks failed=0
```

aka

TBB

```
Size of struct struct MPHIT hit ☐ = 691200000
Warming up
Launching
setup time time=6.189000 (s)
done ntracks=9600000 tot time=3.128342 (s) time/trk=3.258690e-07 (s)
formatted 10 100 9600 32 300 3.128342 0 6.189000 1
track x avg=6.443559 std/avg=0.000039
track y avg=24.394711 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328098
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
number of tracks failed=0
```

(base) cerati@apollo:~/p2z-tests\$ tail -17 logs/propagate_gcc_alpaka.log

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_gcc_omp.log
produce nevts=100 ntrks=9600 smearing by=1.0e-07
NITER=10
done preparing!
setup time time=4.401000 (s)
done ntracks=9600000 tot time=0.766436 (s) time/trk=7.983708e-08 (s)
formatted 10 100 9600 32 300 0.766436 0 4.401000 64
track x avg=6.443559 std/avg=0.000039
track y avg=24.394711 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328098
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
number of tracks failed=0
```

OMP

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_gcc_kokkosv3.log
Size of struct MPTRK trk[] = 107520000
Size of struct MPTRK outtrk□ = 107520000
Size of struct struct MPHIT hit = 691200000
setup time time=6.183000 (s)
done ntracks=9600000 tot time=1.091332 (s) time/trk=1.136804e-07 (s)
formatted 10 100 9600 32 300 1.091332 0 6.183000 -1
track x avg=6.443559 std/avg=0.000039
track y avg=24.394712 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328098
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
number of tracks failed=0
```

kokkos



Output results - ICC

number of tracks failed=0

produce nevts=100 ntrks=9600 smearing by=1.0e-07 NITER=10 done preparing! setup time time=4.085000 (s) done ntracks=9600000 tot time=0.215275 (s) time/trk=2.242448e-08 (s) formatted 10 100 9600 32 300 0.215275 0 4.085000 64 track x avg=6.443813 std/avg=0.000040 track y avg=24.395778 std/avg=0.000039 track z avg=-80.523956 std/avg=0.000000 track dx/x avg=-0.136014 std=0.000045 track dy/y avg=-0.136370 std=0.000045 track dz/z avg=0.000000 std=0.000000 track pt avg=12.329988 track phi avg=1.325591 track theta avg=2.820849 number of tracks with nans=0

(base) cerati@apollo:~/p2z-tests\$ tail -17 logs/propagate_icc_tbb.log

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_icc_omp.log
produce nevts=100 ntrks=9600 smearing by=1.0e-07
NITER=10
done preparing!
setup time time=4.042000 (s)
done ntracks=9600000 tot time=0.349019 (s) time/trk=3.635615e-08 (s)
formatted 10 100 9600 32 300 0.349019 0 4.042000 64
track x avg=6.443813 std/avg=0.000040
track y avg=24.395778 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136014 std=0.000045
track dy/y avg=-0.136370 std=0.000045
track dz/z avg=0.000000 std=0.000000
track pt avg=12.329988
track phi avg=1.325591
track theta avg=2.820849
number of tracks with nans=0
number of tracks failed=0
```

OMP

```
TBB strict fp
```

TBB

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_icc-strict_tbb.log
produce nevts=100 ntrks=9600 smearing by=1.0e-07
NITER=10
done preparing!
setup time time=11.093000 (s)
done ntracks=9600000 tot time=0.613150 (s) time/trk=6.386979e-08 (s)
formatted 10 100 9600 32 300 0.613150 0 11.093000 64
track x avg=6.443561 std/avg=0.000039
track y avg=24.394716 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328070
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
number of tracks failed=0
```

```
(base) cerati@apollo:~/p2z-tests$ tail -17 logs/propagate_gcc_tbb.log
produce nevts=100 ntrks=9600 smearing by=1.0e-07
NITER=10
done preparing!
setup time time=4.409000 (s)
done ntracks=9600000 tot time=0.554024 (s) time/trk=5.771083e-08 (s)
formatted 10 100 9600 32 300 0.554024 0 4.409000 64
track x avg=6.443559 std/avg=0.000039
track y avg=24.394711 std/avg=0.000039
track z avg=-80.523956 std/avg=0.000000
track dx/x avg=-0.136059 std=0.000045
track dy/y avg=-0.136419 std=0.000044
track dz/z avg=0.000000 std=0.000000
track pt avg=12.328098
track phi avg=1.325565
track theta avg=2.820840
number of tracks with nans=0
 umber of tracks failed=0
```

TBB GCC

Using "-fp-model strict" makes ICC output more similar to GCC, but of course has impact on timing. Bottom line message for the paper is that differences are due to floating point precision and that we should clarify what configuration we use for compilation.



To-do

- Make sure thread configurations are the same or at least optimized for each version
- Re-run timing?
- Check if GPU versions (some of which also run on CPU) need to be changed

